Application Integrity Verification in Confidential Computing Scenario

Enrico Bravi
Politecnico di Torino
Dip. Automatica e Informatica
Torino, Italy
enrico.bravi@polito.it

Silvia Sisinni
Politecnico di Torino
Dip. Automatica e Informatica
Torino, Italy
silvia.sisinni@polito.it

Antonio Lioy
Politecnico di Torino
Dip. Automatica e Informatica
Torino, Italy
antonio.lioy@polito.it

***Abstract*—The proliferation of cloud computing has transformed the deployment and scalability of applications, enabling organizations to leverage virtualized infrastructures for enhanced flexibility and efficiency. However, this shift has also introduced significant security and privacy challenges, particularly concerning the protection of sensitive data during processing. Confidential Computing has emerged as a paradigm to address these concerns by safeguarding data in use through hardware-based Trusted Execution Environments (TEEs). TEEs provide isolated environments that ensure the confidentiality and integrity of code and data, even in the presence of potentially compromised host systems. Despite the advancements in TEE technologies, the heterogeneity among implementations poses challenges for developers aiming to create portable and secure applications. Enarx, an open-source project under the Confidential Computing Consortium, addresses this issue by offering a platform-agnostic framework that abstracts the complexities of various TEE architectures, facilitating the deployment of applications across different environments. While Enarx ensures the attestation of the underlying hardware and its own components, it currently lacks mechanisms to allow remote attestation of user-developed applications deployed and running within the TEE. This paper proposes an extension to the Enarx framework that incorporates a mechanism that enables application-level remote attestation, guaranteeing the trustworthiness of workloads deployed in TEEs. By integrating a Trust Monitor system into the remote attestation process, our approach enables the validation of application authenticity and integrity, thereby strengthening the overall security posture of Confidential Computing deployments. This advancement is particularly pertinent for sectors requiring stringent data protection measures, such as finance, healthcare, and critical infrastructure.**

***Index Terms*—Confidential Computing, Trusted Execution Environment, Trusted Computing, Remote Attestation**

I. Introduction

The evolution of Information and Communication Technology (ICT) infrastructures has transitioned from centralized systems to distributed architectures, with cloud computing emerging as a dominant paradigm [1]. This transformation has facilitated unprecedented scalability and accessibility of computing resources, allowing organizations to outsource the provisioning and management of hardware and software to third-party cloud service providers. Consequently, sensitive data is increasingly processed and stored in environments outside the direct control of data owners, raising significant concerns regarding data confidentiality, integrity, and compliance with privacy regulations [2], [3].

To mitigate these concerns, the concept of *Confidential Computing* has been introduced, focusing on protecting data during processing by leveraging hardware-based Trusted Execution Environments (TEEs) [4]. TEEs provide isolated execution contexts that prevent unauthorized access to code and data, even in scenarios where the host operating system or hypervisor is compromised. Major industry players have recognized the potential of Confidential Computing, leading to the formation of the Confidential Computing Consortium (CCC) under the Linux Foundation [5], which aims to accelerate the adoption of secure computing practices through open collaboration.

Despite the promise of TEEs, the diversity of implementations across different hardware vendors presents challenges for developers, who must tailor their applications to specific TEE architectures. This fragmentation hinders the widespread adoption of Confidential Computing solutions. Enarx addresses this issue by providing a unified framework that abstracts the underlying TEE complexities, enabling developers to deploy applications across various platforms without modification. Enarx ensures the attestation of the host environment and its own components, establishing a foundation of trust for application deployment. However, a critical aspect remains unaddressed: the attestation of the user-developed applications themselves. Without mechanisms to validate the authenticity and integrity of application code deployed in the TEE, there exists a risk of executing tampered or malicious workloads within the TEE, potentially compromising the security guarantees of Confidential Computing.

Paper Contribution. Our research introduces an extension to the Enarx framework that incorporates application-level integrity verification into the remote attestation process. By

This work has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme with Grant Agreement No.101139198 (iTrust6G project). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or SNS-JU. Neither the European Union nor the granting authorities can be held responsible for them. This work was also partially supported by the project SERICS (PE00000014) under the NRRP MUR program, funded by the European Union - NextGenerationEU. This publication is also part of the project PNRR-NGEU which has received funding from the MUR – DM 352/2022.

integrating a Trust Monitor system [6], we enable the remote validation of application code before execution within the TEE, ensuring that only verified and trusted workloads are executed. This enhancement is particularly relevant for sectors handling sensitive data, such as finance, healthcare, and critical infrastructure, where the assurance of application integrity is paramount.

Paper Structure. The remainder of this paper is structured as follows: Section II provides background information on Confidential Computing and TEEs. Section III briefly presents some works relevant to the topic taken into consideration. Section IV presents the design and implementation of our proposed extension for application integrity verification. Section V concludes the paper with insights into future work and research directions.

II. Background

A. Trusted Computing and Platform Integrity

Trust, within the context of computing, pertains to the predictability and correctness of a platform's behavior. A computing platform is considered *trusted* if it provides a set of elements, called *Roots of Trust*, that allow to verify that it consistently operates according to predefined expectations. Such *Trusted Platforms* (TPs) serve as foundational elements upon which complex, trustworthy systems can be reliably constructed. The Trusted Computing Group (TCG), a global consortium dedicated to advancing computing security, has defined specifications and endorsed technologies related to trusted computing [7]. TCG introduces the concept of a Trusted Platform (TP) characterized by its capability to perform comprehensive integrity measurements of hardware and software components. These measurements typically consist of cryptographic hashes, employing algorithms such as SHA-256, derived from the component's executable code or configuration data. To securely store and manage these measurements, TCG proposes utilizing a dedicated hardware component known as the Trusted Platform Module (TPM) [8]. The TPM, commonly implemented as a cryptographic coprocessor integrated onto the platform's motherboard, functions as a hardware Root of Trust (RoT) [9]. It securely stores integrity measurements in Platform Configuration Registers (PCRs) and provides a secure mechanism for external entities to obtain and verify these measurements.

Assessment of a TP follows the principle of *transitive trust*. This mechanism relies on leveraging trust established in one component to validate the trustworthiness of subsequent components that assume control during the platform boot process. This approach forms a *chain of trust*, systematically verifying each boot phase to ensure the platform initializes into a verified and secure state. Upon successful validation at the system boot level, the integrity verification process can extend upward to operating system components and applications, extending the chain of trust to the system runtime. Remote entities can subsequently evaluate the trustworthiness of such platforms by employing Remote Attestation (RA) protocols.

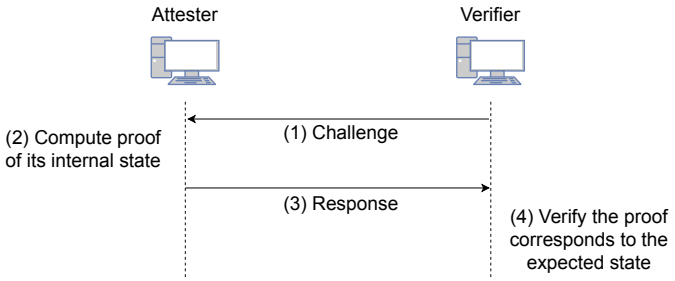

Fig. 1. Simplified workflow of the Remote Attestation process.

B. Remote Attestation

Remote Attestation (RA) is a security protocol enabling a trusted entity, known as the *Verifier*, to assess the integrity and trustworthiness of a remote platform, referred to as the *Attester* or *Prover*. This mechanism is pivotal in distributed computing environments, ensuring that remote systems operate in expected and uncompromised states. The RA process, shown in Fig. 1, operates on a challenge-response model. Initially, the Verifier, possessing knowledge of the Prover's expected state, issues a challenge to the Prover. Upon receiving this challenge, the Prover generates an *Integrity Report* (IR) comprising attestation data reflecting its current state and a signature over both the attestation data and the challenge. The IR is sent back as a response to the Verifier. The Verifier, upon receiving the IR, compares the attestation data against the known expected state. A match indicates that the Prover is in an expected and trustworthy state. This protocol is fundamental in scenarios where verifying the integrity of remote systems is essential, such as in cloud computing and distributed networks.

C. Trusted Execution Environments

In the realm of information technology, a privacy-by-design approach is paramount for safeguarding user data. While various Privacy-Enhancing Technologies (PETs) have been proposed [10], Trusted Execution Environments (TEEs) have gained prominence for protecting data during processing. A TEE [11] is a secure and isolated execution environment within a device, leveraging hardware-based mechanisms to ensure:

- *Authenticity*: Verifying that only authorized code executes within the TEE.
- *Integrity*: Ensuring that the code and data remain unaltered during execution.
- *Confidentiality*: Protecting code and data from unauthorized access.

Some TEEs provide Remote Attestation capability, allowing external parties to verify the TEE's integrity and trustworthiness. The specific threat model addressed by a TEE varies based on its design and implementation, but the overarching goal is to provide strong isolation from the main operating system and other applications.

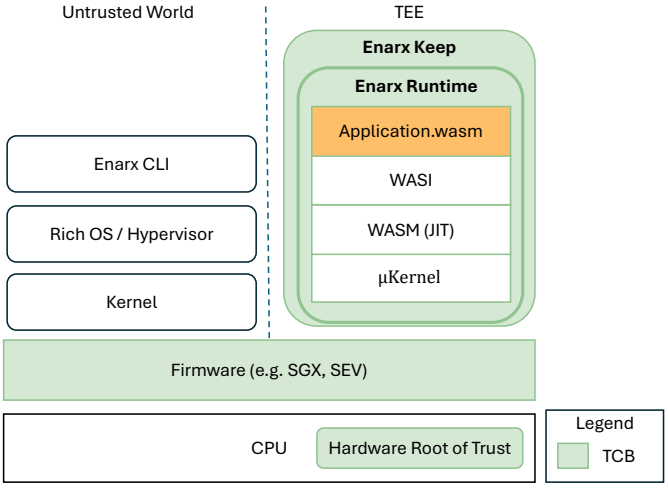

Fig. 2. Enarx components and Trusted Computing Base (TCB).

Several organizations have developed proprietary TEE solutions, such as Intel's Software Guard Extensions (SGX) [12], AMD's Secure Encrypted Virtualization (SEV) [13], and Intel's Trust Domain Extensions (TDX) [14]. However, the lack of standardized interfaces and APIs across different TEEs poses challenges for widespread adoption. Efforts like the GlobalPlatform's TEE specifications [15] and the Trusted Execution Environment Provisioning (TEEP) [16] aim to address these interoperability issues.

D. Enarx: a TEE-Agnostic Solution to Protect Data in Use

Enarx is an open-source project designed to abstract the complexities of various TEE implementations, enabling developers to deploy applications without tailoring them to specific TEEs. By leveraging WebAssembly (WASM) [17], Enarx ensures that applications are portable and can run across different hardware architectures. Currently, Enarx supports Intel SGX2 [18] and AMD SEV-SNP [13] TEEs. Its architecture (Fig. 2) emphasizes minimalism, adhering to the principle of a minimal Trusted Computing Base (TCB) [19]. This design choice reduces the potential attack surface and enhances security.

The core components within the Enarx *Keep*, i.e. the isolated execution environment, include:

1) *Loader* or *Virtual Memory Manager* (VMM): Manages memory allocation and process isolation within the Keep.
2) *Linux Microkernel* (Shim): Provides essential kernel functionalities in a minimal footprint.
3) *WebAssembly Runtime*: Executes applications compiled to the WASM format.
4) *WebAssembly System Interface* (WASI): Offers standardized APIs for system-level operations, ensuring compatibility and ease of development.

D.1. The Steward

Within Enarx's architecture, the *Steward* plays a critical role in the attestation process. Acting as a Remote Attestation service, the Steward verifies the integrity of the hardware and the Enarx Keep before allowing application execution. This aligns with the principles of Zero Trust Architecture, where trust is never assumed and must be continuously verified [20]. Key features of the Steward include:

- *Modularity*: Capable of processing attestation evidence from various hardware platforms, accommodating the diverse architectures of different TEEs.
- *Pluggability*: Supports the integration of new evidence types and the addition of support for emerging TEEs.
- *Scalability*: Designed to handle numerous attestation requests efficiently, making it suitable for large-scale deployments.

Beyond verification, the Steward also functions as a Certification Authority. It interprets vendor-specific attestation evidence and issues standardized public key certificates (e.g., X.509), facilitating trust establishment in broader systems and networks.

III. Related Work

In the evolving domain of confidential computing, several open-source initiatives have emerged alongside Enarx, each aiming to enhance data protection during processing and to facilitate the adoption of TEEs across diverse platforms. These projects, often under the auspices of the Confidential Computing Consortium (CCC) [5], strive to establish standardized, interoperable frameworks that bolster trust in cloud and edge computing infrastructures, contributing to the maturation of confidential computing.

Gramine [21] is an open-source library operating system (LibOS) designed to enable the execution of unmodified Linux applications within Intel SGX enclaves. By providing a lightweight LibOS layer, Gramine allows applications to run in isolated environments without necessitating significant code modifications. This approach simplifies the transition of existing applications to secure enclaves, thereby promoting the adoption of confidential computing practices. Gramine supports native Linux binaries and is compatible with Intel SGX-enabled platforms, offering a practical solution for securing applications in untrusted environments.

Occlum [22] presents a memory-safe, multi-process LibOS tailored for Intel SGX. Written in Rust, Occlum emphasizes security and reliability, reducing the likelihood of memory-related vulnerabilities. It facilitates the execution of unmodified applications within enclaves through a user-friendly interface, requiring minimal configuration. Occlum supports efficient multitasking, multiple filesystems, and provides built-in toolchains to ease application porting. Its design prioritizes ease of use and performance, making it suitable for deploying secure applications in cloud environments.

Veraison [23] is an open-source project focused on constructing components for attestation verification services. Recognizing the diversity of attestation formats across different TEEs, Veraison offers a flexible and extensible framework that

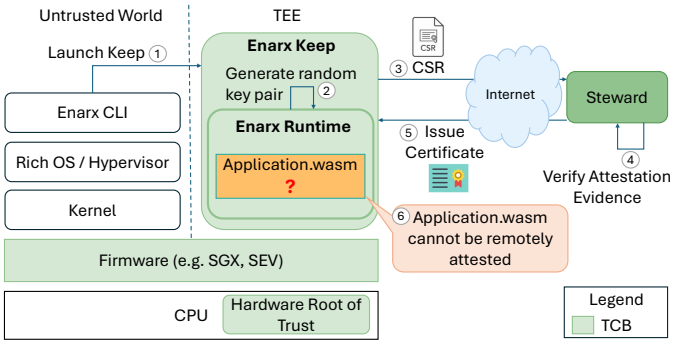

Fig. 3. Overall flow of Enarx deployment and Keep attestation without extending the measurement chain to the WASM application.

can be adapted to various deployment scenarios. It provides a core verification structure and provisioning pipelines, which can be extended through plugins to support specific attestation technologies. Veraison aims to simplify the development of verification services, ensuring that attestation evidence can be accurately and efficiently evaluated.

IV. Solution Design

A. Threat Model

In cloud computing scenarios, establishing mutual trust between cloud service providers (Hosts) and their clients (Guests) is critical yet inherently challenging. Enarx addresses this challenge by creating a secure execution environment in which neither party is required to implicitly trust the other [24].

Host Perspective: The Host (cloud provider) must ensure the integrity, confidentiality, and availability of its infrastructure against potential malicious actions from Guests. Key threats include privilege escalation, resource exhaustion (such as denial-of-service attacks), unauthorized access to sensitive infrastructure resources, and threats to multi-tenant isolation. To mitigate these risks, the Host retains full authority over resource allocation, usage monitoring, and enforcement of restrictions or termination of guest workloads when necessary.

Guest Perspective: The Guest requires strong guarantees of confidentiality, integrity, and execution correctness for both the data processed and the associated application code. Given that Hosts may be partially compromised, malicious, or administered by untrustworthy personnel, Guests cannot solely rely on assurances provided by the Host. Hence, Guests require an independent and verifiable mechanism to validate the integrity of their workload and execution environment before execution begins.

Enarx addresses these threats by leveraging hardware-backed TEEs to provide isolation through secure execution contexts called *Keeps*. Cryptographic attestation mechanisms ensure that the Keep's software and hardware components are trustworthy prior to the execution of the Guest workload.

In the context of our implementation, we specifically consider the scenario where an adversary may attempt to compromise the workload by tampering with WebAssembly (WASM) binaries before their execution. Such tampering could lead to malicious actions compromising the confidentiality, integrity, and reliability of the Guest's computations. We assume the adversary cannot physically tamper with the Host's CPU, memory, or bus; hence, the security assurance primarily depends on the Enarx framework and the underlying hardware TEE capabilities. Furthermore, the Host is assumed to be semi-trusted. While adequate procedures, technical safeguards, and security policies are assumed to be in place, malicious insiders or compromised administrative accounts could still pose significant risks. To mitigate these risks, our implementation extends Enarx by integrating with a Trust Monitor capable of performing a remote integrity check on WASM applications prior to their execution within the Keep. This enables proactive detection of maliciously modified workloads, prevents execution of compromised code, and ensures auditable records of integrity verification.

B. Security Requirements

Based on the defined threat model, the following security requirements are established to guide the integration of the Trust Monitor with Enarx:

- *(SR1) Remote Attestation*: The system shall support cryptographic remote attestation of workload integrity, enabling Guests to independently verify the trustworthiness of workloads prior to execution.
- *(SR2) Integrity Verification*: The Trust Monitor shall verify the cryptographic integrity of WASM binaries before execution within a Keep, ensuring that workloads have not been maliciously altered.
- *(SR3) Isolation Enforcement*: Enarx must enforce strict isolation between workloads executed in different Keeps, preventing cross-tenant attacks or unauthorized access to sensitive data.
- *(SR4) Auditability*: All integrity verification events and attestation outcomes must be securely logged, timestamped, and protected from tampering, ensuring a reliable audit trail.

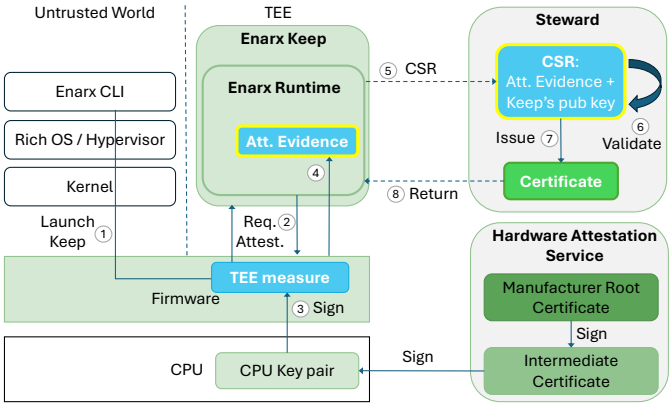

Fig. 4. Overall creation of attestation report, CSR and, following validation by the Steward.

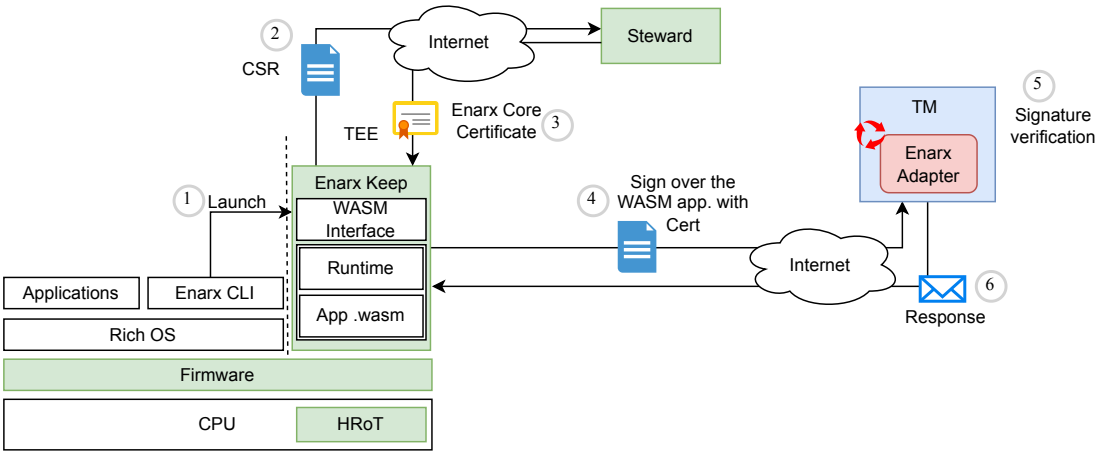

Fig. 5. Overall flow of Enarx deployment and Keep attestation without extending the measurement chain to the WASM application.

- *(SR5) Revocation and Response*: The system shall provide mechanisms to revoke trust and halt execution immediately if integrity checks fail or if malicious activities are detected.
- *(SR6) Minimal Trust Assumption*: The attestation and verification process must rely exclusively on hardware-backed TEE and cryptographic primitives, minimizing trust assumptions related to the Host's infrastructure.
- *(SR7) Resistance to Insider Threats*: The solution shall incorporate safeguards against insider threats, ensuring that administrative access or internal compromise of Host systems cannot invalidate or bypass workload attestation and integrity verification mechanisms.

C. Enarx Remote Attestation Process

Enarx's primary goal is to ensure the principles of Confidential Computing, specifically focusing on the integrity and confidentiality of application code and data. This assurance is realized by relying on TEEs properties and remote attestation capabilities.

The complete workflow for generating and validating the attestation report and CSR is summarized in Fig. 4. Upon execution via the Enarx Command Line Interface (CLI), a request is initiated to establish a new TEE on the underlying hardware (Host) (1). Subsequently, the Enarx core is loaded into the TEE, creating a secure Keep. At this stage, the attestation process begins (2). Rather than directly measuring the Keep, Enarx gathers attestation data from the Host CPU firmware and forwards it to a trusted entity known as the Steward.

Due to variability across TEE technologies, obtaining reliable and consistent CPU measurements is inherently challenging. The CPU measurement, provided by the Host firmware upon request, encompasses critical hardware-specific information including CPU architecture details (such as bit length, model, stepping, and firmware version). However, the CPU measurement alone is inadequate due to potential replay attacks. To mitigate this, additional contextual information is integrated into the attestation report, including the hash of the Enarx Keep's initial state (prior to workload deployment) and the hash of a randomly generated signing key used for authenticating the Enarx keep. These supplementary details bind the attestation report uniquely to the specific Enarx deployment instance. The finalized attestation report is cryptographically signed using a CPU-generated key pair responsible for creating the TEE. Trust in this key pair by the Steward relies on validation through a hierarchical certificate chain, ultimately rooted in a certificate provided out-of-band by the CPU hardware manufacturer.

Upon signing, the attestation report is embedded as an extension within a Certificate Signing Request (CSR) in PKCS10 format, along with the public key of the Enarx Keep. This CSR is securely transmitted to the Steward via an HTTPS channel, leveraging a pre-established TLS certificate for secure communication (5). Successful attestation by the Steward (6) results in the issuance of a certificate for the Enarx Keep public key (7). The certificate, which represents the trustworthiness of the Host's TEE hardware and the Enarx Keep's runtime, is returned to the Enarx Keep (8) and constitutes a prerequisite for deploying the application's workload inside the Keep.

D. Enarx Extension

To introduce the possibility of verifying the integrity of an application in Enarx, the current attestation flow must be extended. In particular, this procedure has to be placed before the requested payload is executed, in order to allow it only if it results as not compromised. This verification has to be placed after the Keep requests its credentials from the Steward. Once these credentials are obtained, the Enarx components inside the Keep can be considered trustworthy, and for this reason, they can be leveraged for verifying the application code, allowing for building a chain of trust among the Keep components. The proposed schema is depicted in Fig. 5. The integrity verification for applications is integrated with an external component from Enarx: The Trust Monitor (TM).

This system enables flexible and extensible RA procedures for heterogeneous infrastructures.

The process follows the current Enarx procedure, starting with the execution request of a Keep (1). Once the Keep is instantiated, before starting the execution, it requests its credentials from the Steward, sending a CSR containing the integrity evidence for the Enarx components (2). The Steward verifies the validity of the CSR (Keep public key and attestation evidence), and if it succeeds, it sends to Keep its certificate (3). At this point, the application integrity verification is placed because the execution of the Keep has not yet begun. Once the Keep receives its credentials, it performs an integrity measurement over the request workload code and sends it to the TM for verification (4). The Enarx adapter, in charge of managing the communication with the Enarx framework, manages the request and evaluates the integrity of the application. Once the Enarx adapter produces a result, it returns it to the Keep (5). In this case, the result can be successful or not. If the result is successful, then the Keep allows the application execution, and the workload requested is deployed and executed. If the result reports a workload compromised, the Keep aborts its execution and stops the procedure.

The TM allows for more flexible management of the attestation procedure and the auditing logging. Being an external component from the Enarx architecture permits to introduce very few changes in the Enarx core. This also introduces flexibility on the verification side with the Enarx adapter, which is an extensible module that can be adapted to all the TEE technologies eventually supported by Enarx.

V. Conclusion and future works

This paper presented an extension to the Enarx framework, enhancing its capabilities in remote attestation by enabling comprehensive integrity verification of user workloads before executing inside a Keep. The proposed extension integrates seamlessly with a TM and permits to extend the chain of trust for measurement up to the user workload. Our proposal satisfies the previously defined security requirements. Firstly, by performing remote attestation (SR1) using the Enarx Keep credential, the extended Enarx framework ensures the integrity and authenticity of workloads before execution. The TM verifies the integrity of WASM binaries by comparing the attestation report against trusted measurements stored in its whitelist database (SR2). Isolation enforcement (SR3) remains inherently provided by Enarx's secure Keep environment, effectively mitigating risks related to cross-tenant interference. Additionally, the TM securely logs all attestation events and results, ensuring comprehensive auditability (SR4). The extension also includes robust revocation and response capabilities, allowing for immediate remediation upon detecting integrity violations (SR5). By strictly relying on secure cryptographic primitives performed inside hardware-back TEE, the solution does not rely on potentially compromised host infrastructure, satisfying minimal trust assumptions (SR6). Furthermore, the architecture inherently protects against insider threats, as workload attestation process cannot be circumvented through administrative privileges or host system compromise (SR7), without being detected by the TM.

The current approach provides a robust basis for remote attestation within Enarx, though future enhancements could involve aggregating Keep-level and workload-level attestation reports within the TM for streamlined verification and simplified management. As future work, a PoC implementation is planned to verify the effectiveness and correctness of the proposed solution design.

References

[1] R. Rai, G. Sahoo, and S. Mehfuz, "Exploring the factors influencing the cloud computing adoption: a systematic study on cloud migration", Springerplus, vol. 4, 2015, pp. 1–12, doi: 10.1186/s40064-015-0962-2

[2] V. Raja and B. Chopra, "Exploring challenges and solutions in cloud computing: A review of data security and privacy concerns", Journal of Artificial Intelligence General Science (JAIGS), vol. 4, April 2024, p. 121–144, doi: 10.60087/jaigs.v4i1.86

[3] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing", IEEE Communications Surveys & Tutorials, vol. 15, no. 2, 2013, pp. 843–859, doi: 10.1109/SURV.2012.060912.00182

[4] Global Platform, "Introduction to Trusted Execution Environments", May 2018, https://globalplatform.org/wp-content/uploads/2018/05/Introduction-to-Trusted-Execution-Enviroment-15May2018.pdf

[5] The Confidential Computing Consortium, https://confidentialcomputing.io/

[6] E. Bravi, D. G. Berbecaru, and A. Lioy, "A Flexible Trust Manager for Remote Attestation in Heterogeneous Critical Infrastructures", 2023 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Naples (Italy), December 2023, pp. 91–98, doi: 10.1109/CloudCom59040.2023.00027

[7] The Trusted Computing Group, https://trustedcomputinggroup.org/

[8] TCG, "TPM main part 1 design principles", https://trustedcomputinggroup.org/wp-content/uploads/TPM-Main-Part-1-Design-Principles v1.2 rev116 01032011.pdf

[9] Trusted Computing Group, "Root of Trust", TCG glossary

[10] Information Commissioner's Office, "Privacy-enhancing technologies (PETs)", 2023, https://ico.org.uk/media/for-organisations/uk-gdpr-guidance-and-resources/data-sharing/privacy-enhancing-technologies-1-0.pdf

[11] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted execution environment: What it is, and what it is not", 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki (Finland), December 2015, pp. 57–64, doi: 10.1109/Trustcom.2015.357

[12] W. Zheng, Y. Wu, X. Wu, C. Feng, Y. Sui, X. Luo, and Y. Zhou, "A survey of intel sgx and its applications", Frontiers of Computer Science, vol. 15, 2021, pp. 1–15, doi: 10.1007/s11704-019-9096-y

[13] Advanced Micro Devices, Inc., "Secure Encrypted Virtualization API Version 0.24", Tech. Rep. Publication #55766, Revision 3.24, April 2020

[14] P.-C. Cheng, W. Ozga, E. Valdez, S. Ahmed, Z. Gu, H. Jamjoom, H. Franke, and J. Bottomley, "Intel tdx demystified: A top-down approach", ACM Comput. Surv., vol. 56, April 2024, doi: 10.1145/3652597

[15] Global Platform, "TEE System Architecture v1.3 (GPD_SPE_009)", May 2022, https://globalplatform.org/specs-library/tee-system-architecture/

[16] M. Pei, H. Tschofenig, D. Thaler, and D. Wheeler, "Trusted Execution Environment Provisioning (TEEP) Architecture", RFC-9397, July 2023, doi: 10.17487/RFC9397

[17] The WebAssembly Project, https://webassembly.org/

[18] V. Costan and S. Devadas, "Intel SGX Explained", Cryptology ePrint Archive, Paper 2016/086, 2016, https://eprint.iacr.org/2016/086

[19] M. Nieles, K. Dempsey, V. Y. Pillitteri, *et al.*, "An introduction to information security", NIST SP800-12, 2017, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-12r1.pdf

[20] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture", NIST SP800-207, August 2020, doi: 10.6028/NIST.SP.800-207

[21] The Gramine project, https://gramineproject.io/

[22] The Occlum project, https://occlum.io/

[23] The Veraison project, https://github.com/veraison

[24] Enarx, "Threat Model", https://enarx.dev/docs/technical/threat